Simulating High-Level Robot Behaviors

東京大学情報工学専攻　○ カフナー・ジェームズ　加賀美 聡　稲葉 雅幸　井上 博允

The University of Tokyo　○ J. KUFFNER　S. KAGAMI　M. INABA　H. INOUE

Abstract : Physically-based simulation software is commonly used for developing and testing low-level robot control algorithms. In order to facilitate the development and evaluation of higher-level robot behaviors, broader-based simulations are needed. Examples include software for simulating 3D vision, motion planning for obstacle avoidance, and integrating vision and planning. In addition to modeling the general interaction between the robot and its environment, the software can potentially be used as a graphical user interface for directly controlling or interacting with a robot operating in the real world. This paper describes our current efforts toward building a large-scale software simulation framework for the development and testing of high-level behaviors for humanoid robots. We view this as a potential useful tool for the visualization and development of robotic systems, as well as an interactive, task-level programming interface for robots.

1 Introduction

Much research effort in robotics has been focused on designing high-level control architectures for autonomous robots [1–3]. The primary aim of our simulation environment is to assist in the development and testing of such high-level control software. For humanoid robots, we are interested in developing software to solve complex tasks, such as navigating in an unknown environment while avoiding obstacles, and grasping and manipulating objects.

Advances in computing hardware have enabled increasingly sophisticated software simulations to be used in the design and testing of robotic systems. In particular, complex robots such as humanoid robots have benefited from dynamic simulation software. Such software has assisted in the realization of dynamic walking in several humanoid robots [4–6].

In comparison, much less attention has been devoted to simulating higher-level behaviors for humanoid robots. Successful simulation environments have already been developed for teleoperated space robots [7–9]. Similar kinds of software capabilities are needed for humanoid robotic systems, in particular software libraries for 3D vision, motion planning for obstacle avoidance, and integrating vision and planning are needed. To facilitate the deployment of such software, we are currently developing a graphical simulation environment for the H5 dynamic humanoid robot (see Figure 1 and Figure 3). The project builds upon a software framework that was originally developed for the high-level control of computer animated characters in 3D virtual environments [10]. Although its present capabilities are limited, it is our hope that through the use of such interactive simulation software, the current and future capabilities of humanoid and other complex robotic systems can be improved.

The paper is organized as follows: Section 2 gives an overview of the simulation environment and its larger goals, along with a brief description of two components: simulated vision (Section 2.2) and motion planning (Section 2.3). Section 3 shows some preliminary experimental results using simulated vision for perception-based global navigation for a humanoid robot, and path-planning for object manipulation. Section 4 concludes with a summary discussion.

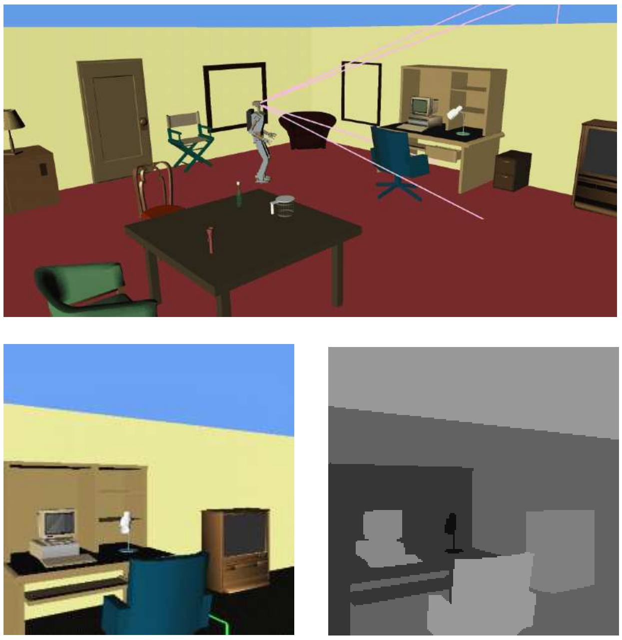

Fig. 1: Simulated vision of a humanoid robot navigating in a virtual office.

2 Simulation Environment

2.1 Overview

For simulation purposes, we are designing a system architecture similar to the *TeleSensorProgramming* architecture proposed in [7]. We begin by constructing a *virtual robot* in a virtual environment. The virtual robot comes equipped with *virtual sensors* and *virtual control inputs*. Any software being developed for controlling a real robot can first be tested and evaluated by connecting its inputs and outputs to the virtual robot.

The ultimate goals of the simulation environment are:

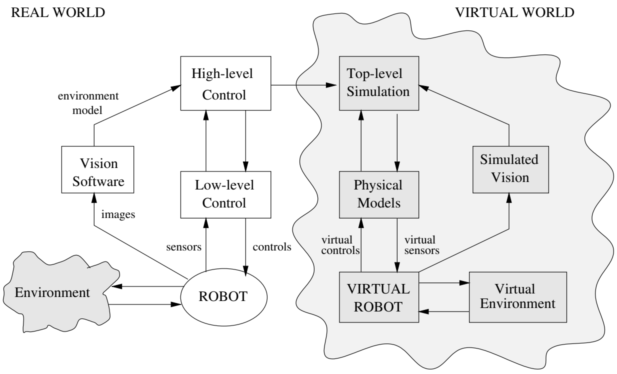

Fig. 2: Simulation architecture overview.

1. Visualize the robot model and movement in a simulated 3D environment.

2. Provide a testbed for the development and evaluation of robot control and behavior software.

3. Serve as an interactive, task-level user interface for controlling a real robot.

4. Provide a means for a robot to reason about the physical environment.

By "reasoning" about the environment, we mean that the robot has the ability to *compute the potential consequences of a set of possible actions*. This will hopefully allow the robot to make better informed decisions when attempting to solve complex tasks.

Clearly, a virtual robot has many advantages over a real robot. In particular, if one so desires, one can equip a virtual robot with "perfect" sensing and "perfect" control. However, this is typically not a realistic scenario. One of the primary advantages to using a flexible simulation environment is that one has the freedom to construct detailed models of sensor noise, or errors in control. By testing control strategies under different conditions, one can potentially distinguish between failure of the algorithm, or failure due to sensor noise or control errors.

The key point is that one can build software models that are better than current robot technology. Thus, while the designs and technology used in real robots improves in the future, the software to control them can be developed at the present time.

The simulation environment separates high-level and low-level control, and provides a model of robot-environment interaction (see Figure 2) If developed properly, the simulation could provide the robot with the means to anticipate the consequences of a set of possible actions. For example, suppose a humanoid robot is attempting to pick up a nearby object. If the robot has general knowledge of the physical properties of the object and its current situation in the environment, simulation can be used to help decide the best way to accomplish the given task. The criteria used might consist of the best way to grasp, lift, and transport the object given the physical limitations of the robot and the object's properties. In this way, a robot can attempt to select a plan of action that will minimize the chance of injury or error, while maximizing the realization of its goals and tasks.

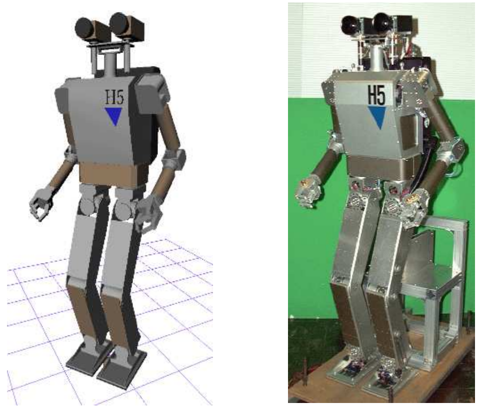

Fig. 3: The virtual and real dynamic humanoid 'H5'.

In developing our simulation environment for the H5 dynamic humanoid, we have initially focused on the specific tasks of using simulated vision for perception-based navigation (using bipedal locomotion), and path planning for object grasping and manipulation. Our goal is to build a series of abstraction layers between the fine-grained, low-level control of the motors [11], and the robot's high-level goals.

2.2 Simulated Vision

One of the long-term goals of the robotics research community is to build reliable software that will allow a robot to explore an unknown environment, and incrementally build its own internal model of the world through sensors. Toward this purpose, we are developing a simulated vision system for testing and evaluating experimental vision algorithms. By utilizing graphics rendering hardware, a virtual robot can acquire virtual camera images from a 3D model of a scene.

Computer graphics and computer vision are complementary disciplines: while the goal of graphics is to *generate images from a model*, the goal of vision is to *extract a model from images*. In order to evaluate the efficacy of an experimental vision technique, the following steps can be performed:

1. The virtual robot acquires image data from a scene (the scene is graphically rendered from the robot's

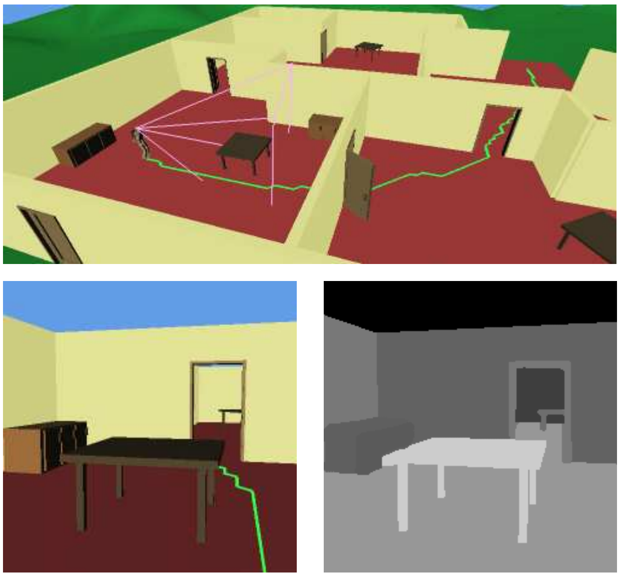

Fig. 4: Simulated view of a humanoid robot.

point of view).

2. An internal model of the sensed virtual environment is constructed using whatever experimental vision technique is being tested.
3. The internal model is directly compared to the actual 3D model of the scene and an appropriate error metric is computed.

We are currently developing a simulated vision system for testing and evaluating a real-time 3D vision algorithm that has been used for legged robot navigation [12]. Similar simulated vision systems have also been successfully used in teleoperation applications [7].

A simulation environment can also be used for testing vision techniques other than those used for model-building. For example, evaluating image segmentation or object recognition techniques, the scene model can be initially divided up into a collection of small to medium-sized object models. Each object model is assigned a unique ID. A color table is initialized to represent a one-to-one mapping between object IDs and colors. To check which objects are visible to a virtual robot, the scene can be rendered from the robot's point of view, using flat shading and using the unique color for each object as defined by the object ID. If the resulting image pixels are scanned, a list of visible objects can be obtained from the pixel color information and compared to the output of the vision algorithm.

2.3 Motion Planning

Motion planning problems involve searching the system configuration space for a collision-free path connecting a given start and goal configuration [3]. For problems in low dimensions, the configuration space can often be explicitly represented and complete algorithms are known [13, 14]. For high-dimensional configuration spaces however, it is typically impractical to explicitly represent the configuration space. Instead, the space is usually randomly sampled in order to discover free configurations [15].

In the context of humanoid robots, explicit motion planning techniques can be used for global navigation, while randomized sampling techniques are more appropriate for computing object manipulation motions or body posture tasks.

We have initially focused on motion planning for two types of tasks: global navigation among obstacles, and computing collision-free object manipulation motions. For navigation, the task of the robot is to move from a start location to a goal location while avoiding obstacles. For object manipulation, the task of the robot is to successfully grasp and reposition a physical object in the environment without colliding with other obstacles. Through sensing, the robot builds a model of the environment which it uses for motion planning.

For navigation planning, an explicit representation of the configuration space may be used to plan a single path from a starting location to a given goal location. For manipulation planning, the task is specified by identifying a target object to be moved and its new location. The motion planning software will then attempt to compute three trajectories:

- *Reach*: Move the robot in position to grasp the object.
- *Transfer*: After grasping, move the object to the target location
- *Return*: Once the object has been placed at the target location, release it and return the robot to its rest position.

If a path at each phase is returned by the path planner, the robot executes the motion, possibly guided by visual or force feedback.

There are many potential uses for such software, with the primary one being a high-level control interface for solving complex navigation or object manipulation tasks. Some basic experiments using path planning in a simulation environment for both manipulation and global navigation are described in the following section.

3 Experiments

This section presents some preliminary results using a prototype simulation environment. Although the current capabilities are limited, our goal is to incrementally build upon the existing software framework for future experiments.

3.1 Perception-Based Navigation

Robust sensor-based navigation algorithms are vital for the development of humanoid robots that can coexist with real humans. Figure 1 and Figure 4 depict a humanoid robot navigating in a virtual office. An outline of a representative portion of the robot's viewing frustum is shown in each. The bottom two images show the scene rendered from the robot's point of view, the left image being full-color, and the right image an object-based segmentation image.

Using the simulated vision technique described in Section 2.2 and a fast navigation path planner [10], the behavior of a virtual robot was tested while exploring an unknown maze environment. Figure 5 shows snapshots at various stages of the interactive simulation. The final path is solely the result of the interaction between path planning based in the robot's internal model and the visual feedback obtained during exploration.

Using simulation, the overall behavior of the virtual robot can be visualized and tested under a variety of circumstances in many different environments. The primary goal of this scheme is to assist in the development of robust high-level behavior control software which can eventually be used to control the corresponding real robot.

3.2 Perception-Based Grasping and Manipulation

The simulation environment can be used not only for tasks involving navigation, but for a variety of tasks that require visual coordination. This includes object grasping and manipulation.

In the examples shown in Figure 6 and Figure 7, the synthetic vision module is used in order to verify that a particular target object is visible to a virtual humanoid prior to attempting to grasp it. If the object is visible, a manipulation planner is invoked to plan a collision-free path to grasp the object. If the target object is not visible, the humanoid will attempt to reposition itself, or initiate a searching behavior in an attempt to find the missing object.

The path planning algorithm used is *RRT-Connect* [16], a randomized algorithm that was originally developed to plan collision-free motions for animated characters in 3D virtual environments [10]. Combined with an inverse kinematics algorithm, the planner facilitates a task-level control mechanism for planning manipulation motions. Through a graphical user interface, an operator can click and drag an object to a target location and issue a *move* command.

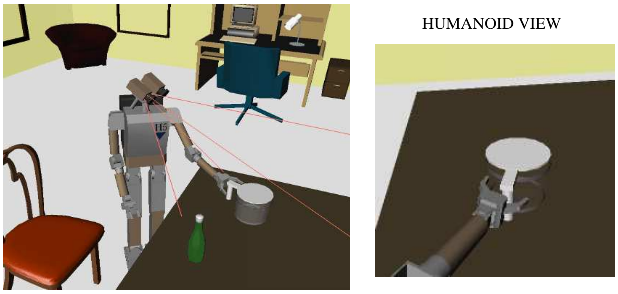

Fig. 6: Reaching and grasping a coffee pot.

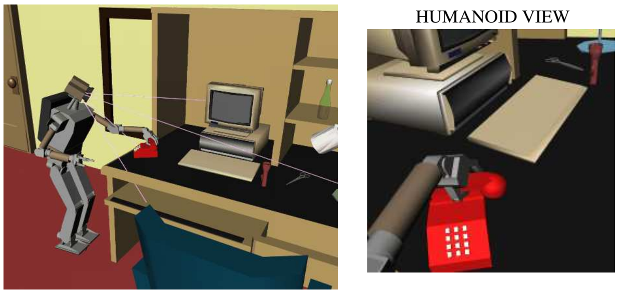

Fig. 7: Answering the telephone.

Figure 8 shows the H5 humanoid model playing chess. Each of the motions necessary to reach, grasp, and reposition a game piece on the virtual chessboard were generated using the RRT-Connect path planning algorithm. Each robot arm comprises a 6-DOF kinematic chain, and the entire scene contains over 10,000 triangle primitives. The 3D collision checking software used for these experiments was the RAPID library based on OBB-Trees developed by the University of North Carolina [17].

4 Discussion

This work proposes using a software simulation environment as a tool for programming future robots. In particular, software for high-level behavior control can be developed either independently or in conjunction with low-level actuator control software. This should help to both shorten the development cycle time, as well as reduce the costs and safety risks involved during the initial phases of testing robotic systems.

In addition, it is our hope that simulation software can ultimately be utilized by robots to anticipate the consequences of their potential actions. In this way, a robot can attempt to select a plan of action that will minimize the chance of injury or error, while maximizing the realization of its goals and tasks.

Many issues remain to be solved. Although simulation software has increased in both sophistication and speed, the results are often highly dependent on the accuracy of

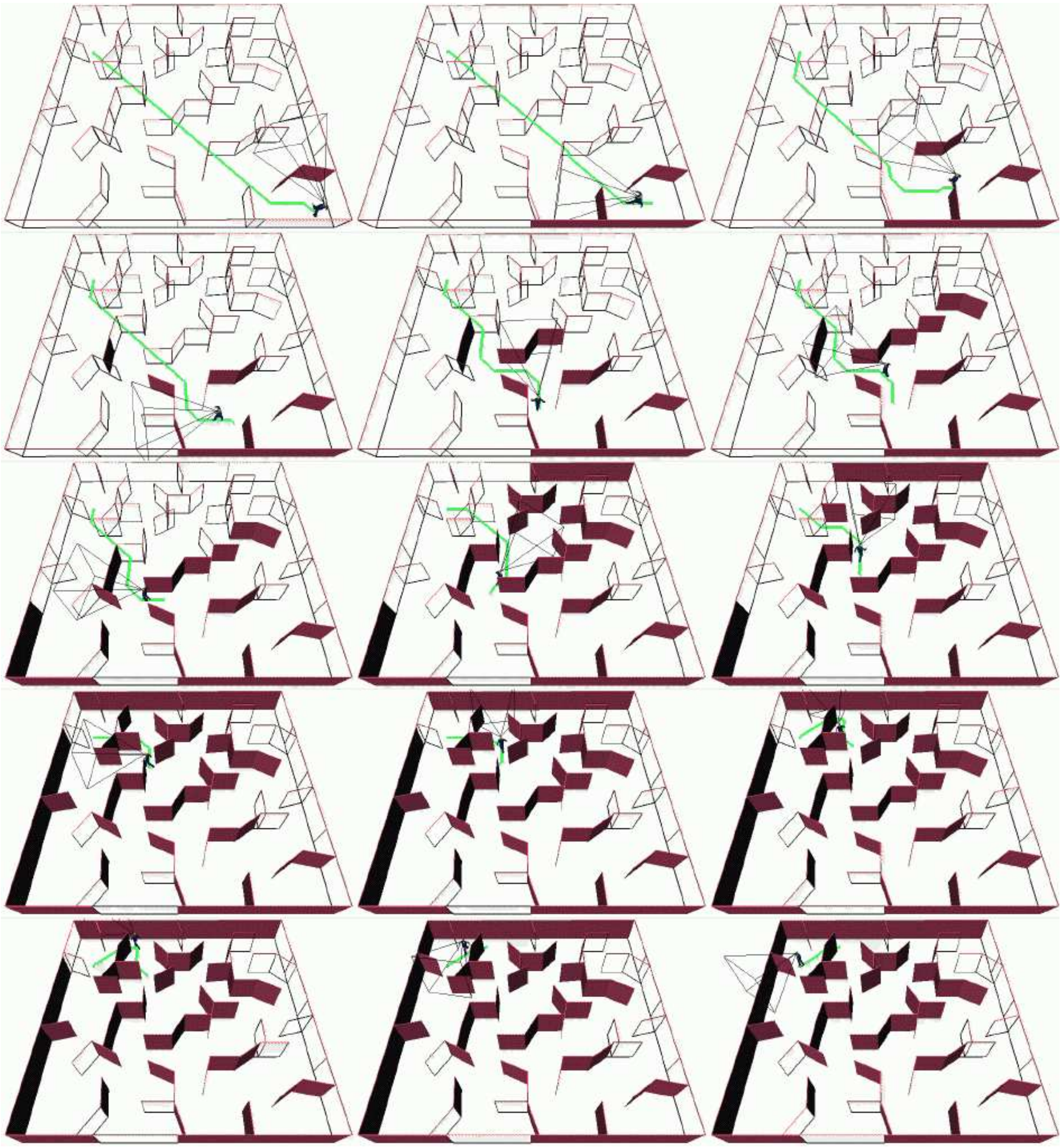

Fig. 5: Snapshots of a virtual robot exploring an unknown maze environment. Information about obstacles in the environment is incrementally added to the robot's internal model of the world as the robot moves. The robot is given the task of navigating from the bottom-right corner to the top-left corner of the maze. A portion of the robot's viewing frustum, along with the current path is shown.

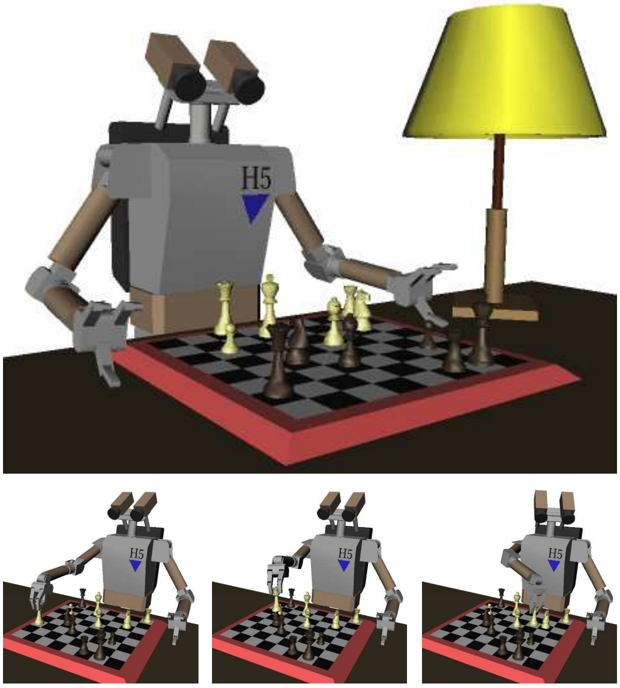

Fig. 8: The H5 humanoid robot playing chess.

the models used. For example, precise dynamic simulation generally requires accurate models of friction. While the use of more complex models improves accuracy, this typically comes at an higher computational cost. However, as computing hardware continues to improve, we can expect to see increasingly sophisticated simulations of robotic systems in the future.

Acknowledgments

This research is supported in part by a Japan Society for the Promotion of Science Postdoctoral Fellowship for Foreign Scholars in Science and Engineering. Many thanks to the Stanford University Dept. of Computer Science, where the initial development of portions of the simulation software was conducted.

References

1) R. C. Arkin. Cooperation without communication: Multiagent schema based robot navigation. *Journal of Robotic Systems*, pp. 351–364, 1992.

2) R. A. Brooks. A layered intelligent control system for a mobile robot. In *Robotics Research The Third International Symposium*, pp. 365–372. MIT Press, Cambridge, MA, 1985.

3) J. C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

4) Kazuo Hirai. Current and future perspective of hona humanoid robot. In *In Proc. of 1997 IEEE/RSJ Int. conf. on Intelligent Robots and Systems (IROS'97)*, pp. 500–508, 1997.

5) J. Yamaguchi, S. Inoue, D. Nishino, and A. Takanishi. Development of a bipedal humanoid robot having antagonistic driven joints and three dof trunk. In *In Proc of 1998 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'98)*, pp. 96–101, 1998.

6) K. Nagasaka, M. Inaba, and H. Inoue. Walking pattern generation for a humanoid robot based on optimal gradient method. In *In Proc. of 1999 IEEE Int. Conf. on Systems, Man, and Cybernetics*, 1999.

7) B. Brunner, K. Landzettel, B.-M. Steinmetz, and G. Hirzinger. Telesensorprogramming - a task-directed programming approach for sensor-based space robots. In *Proc. of Int. Conf. on Advanced Robotics (ICAR)*, 1995.

8) G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl. Sensor-based space robotics - rotex and its telerobotic features. *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 5, October 1993.

9) M. Oda, K. Kibe, and F. Yamagata. ETS-VII - space robot in-orbit experiment satellite. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, April 1996.

10) J. J. Kuffner Jr. *Autonomous Agents for Real-time Animation*. PhD thesis, Stanford University, 1999.

11) Y. Tamiya, M. Inaba, and H. Inoue. Realtime balance compensation for dynamic motion of full-body humanoid standing on one leg. *Journal of the Robotics Society of Japan*, Vol. 17, No. 2, pp. 268–274, 1999.

12) S. Kagami, K. Okada, M. Inaba, and H. Inoue. Large design and development of onbody realtime disparity image generation system. In *In Proc. of 4th Robotics Symposia 1999*, pp. 177–182. Robotics Society of Japan, 1999.

13) J. T. Schwartz and M. Sharir. On the 'piano movers' problem: Ii. general techniques for computing topological properties of real algebraic manifolds. *Advances in applied Mathematics*, Vol. 4, pp. 298–351, 1983.

14) J.F. Canny. *The Complexity of Robot Motion Planning*. MIT Press, Cambridge, MA, 1988.

15) J. Barraquand, L.E. Kavraki, J.C. Latombe, T.Y. Li, R. Motwani, and P. Raghavan. A random sampling scheme for path planning. *Int. J. of Robotics Research*, Vol. 16, No. 6, pp. 759–774, 1997.

16) J.J. Kuffner and S.M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *To appear in Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA'2000)*, San Francisco, CA, April 2000.

17) S. Gottschalk, M. C. Lin, and D. Manocha. Obbtree: A hierarchical structure for rapid interference detection. In *SIGGRAPH '96 Proc.*, 1996.